



**IN THE UNITED STATES PATENT AND TRADEMARK OFFICE**

In re patent application of:

Ken SHOEMAKER

Serial No.: 09/942,812                          Group Art Unit: 2183  
Filed: August 29, 2001                          Examiner: T. Meonske  
FOR: APPARATUS AND METHOD FOR SWITCHING  
THREADS IN MULTI-THREADING PROCESSORS

**APPEAL BRIEF**

Mail Stop Appeal Brief - Patents  
Commissioner for Patents  
P.O. Box 1450  
Alexandria, VA 22313-1450

Sir:

Applicant submits this appeal brief, thus perfecting the notice of appeal filed on April 26, 2005. A petition for a one month extension of time accompanies this brief.

The required headings and subject matter follow.

**(i) Real party in interest.**

This case is assigned of record to Intel Corporation, who is the real party in interest.

**(ii) Related appeals and interferences.**

There are no known related appeals and / or interferences.

**(iii) Status of claims.**

Claims 1-20 are pending in the application. Claims 1-20 stand rejected. The rejections of claims 1-20 are being appealed.

07/29/2005 HVDONG1 00000072 09942812

02 FC:1402

500.00 DP

P10127

**(iv) Status of amendments.**

No amendments have been filed after the final rejection. The attached Claims appendix reflects the current status of the claims.

**(v) Summary of claimed subject matter.**

With respect to claim 1, some embodiments of the invention involve a multi-threading processor, including a first instruction fetch unit (e.g. unit 16, see Fig. 2, page 5, lines 9-17) and a second instruction fetch unit (e.g. unit 18, see Fig. 2, page 5, lines 9-17); a multi-thread scheduler unit (e.g. unit 24, see Fig. 2, page 5, lines 9-17) coupled to the first instruction fetch unit 16 and the second instruction fetch unit 18; an execution unit (e.g. unit 26, see Fig. 2, page 5, lines 9-17) coupled to the scheduler unit 24, wherein the execution unit 26 is to execute a first active thread and a second active thread; and a register file (e.g. file 28, see Fig. 2, page 5, lines 9-17) coupled to the execution unit 26, wherein the register file 28 is to switch one of the first active thread and the second active thread with a first inactive thread.

With respect to claims 9 and 15, some embodiments of the invention involve switching threads in a multi-threading processor, including fetching a first active thread and a second active thread (e.g. block 60, see Fig. 5, page 7, line 27 – page 8, line 1); detecting a stalling event in the first active thread (e.g. block 62, see Fig. 5, page 8, lines 1-2); and switching the first active thread with a third thread, if the third thread is ready to execute (e.g. block 68, see Fig. 5, page 8, lines 5-8).

**(vi) *Grounds of rejection to be reviewed on appeal.***

I. Claims 1-4 and 9-20 stand rejected under 35 U.S.C. § 102(b) as being anticipated by the article Simultaneous Multithreading: A Platform for Next-Generation Processors (hereinafter Eggers).

II. Claims 5-8 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Eggers.

**(vii) *Argument.***

I. The rejection of claims 1-4 and 9-20 under 35 U.S.C. § 102(b) as being anticipated by Eggers is in error and should be reversed.

**Claim 1**

In order to anticipate, the reference must identically disclose each and every claim recitation. Claim 1 recites a first instruction fetch unit and a second instruction fetch unit. Eggers discloses only a single instruction fetch unit that separately fetches two instructions.

The rejection record asserts that Eggers teaches a first instruction fetch unit and a second instruction fetch unit, relying on the assertion that “the fetch unit partitions itself among the threads.” This is a factually inaccurate representation of the teachings of Eggers. The final office action clarifies that the rejection relies on page 14, left column, lines 38-48 of Eggers (reproduced below):

On the other hand, an SMT fetch unit can take advantage of the interthread competition for instruction bandwidth to enhance performance. First, it can partition this bandwidth among the

threads. This is an advantage, because branch instructions and cache-line boundaries often make it difficult to fill issue slots if the fetch unit can access only one thread at a time. We fetch from two threads each cycle to increase the probability of fetching only useful (nonspeculative) instructions. In addition, the fetch unit can be smart about which threads it fetches, fetching only those that will provide the most immediate performance benefit.

Applicants first note that the foregoing paragraph refers to the fetch unit only in the singular (“an SMT fetch unit”, “it”, “the fetch unit”). In other words, only a single fetch unit is described. Moreover, Eggers does not teach that the fetch unit partitions “itself,” as asserted by the office action. Eggers describes that the fetch unit can partition instruction bandwidth, not the fetch unit itself.

In the Advisory Action mailed April 13, 2005, the Examiner admits that the fetch unit does not partition ‘itself’ into more than one fetch unit among the threads and acknowledges that Eggers teaches only partitioning instruction bandwidth among the threads. Accordingly, the rejection of record is admitted to be error and should be reversed. Although applicants submit that the rejection should be reversed for substantive reasons detailed herein, at a minimum the Board should remand the case and require the Examiner to properly articulate the rejection.

In the advisory action, the Examiner then arbitrarily picks and chooses various components internal to the fetch unit for allegedly reading on the claim recitations. This is improper. One skilled in the art, absent the teachings of the present application, simply would not consider that Eggers teaches multiple fetch units. The straightforward teaching of Eggers is a single fetch unit. A single fetch unit with 8 program counters is structurally different from and does not teach or suggest the recited first and second fetch units.

Accordingly, the single fetch unit described in Eggers does not identically describe the recited first and second instruction fetch units. Moreover, a single fetch unit which can partition instruction bandwidth does not identically describe the recited first and second instruction fetch units. Any structural difference between the claims and the cited reference is sufficient to overcome a rejection under § 102. Accordingly, the rejection should be reversed.

Claim 1 further recites a multi-thread scheduler unit coupled to said first instruction fetch unit and said second instruction fetch unit. The office action relies on page 13, left hand column, particularly the description of Fig. 1c, for allegedly identically disclosing this claim recitation. However, the cited portion merely generally describes how one form of SMT might work, and does not describe any particular structure. Applicants further note that the only mention of scheduling is in connection with scheduling machine resources, not scheduling threads. Moreover, the cited portion does not describe a multi-thread scheduler unit coupled to first and second instruction fetch units. In fact, the entire Eggers reference is devoid of any block diagram or other structural description of a fully functional SMT processor. The reference only merely describes different SMT capabilities that are proposed to improve multithreading performance.

The final office action admits that Eggers does not provide any structural description of a fully functional SMT processor. However, the office action asserts that such teaching is not necessary. (See page 4, lines 1-3 of the final office action). The office action relies on an unreasonably broad interpretation of the term “coupled” to make any structural recitation in the claims essentially meaningless. According to the office action, every component which is part of a processor is inherently “coupled” to every other component of the processor. Therefore, according to the office action, even though no structural components or interconnections are actually described in Eggers, every functional component mentioned in Eggers is “coupled” to every other functional component mentioned in Eggers, thereby inherently reading on every possible structural relationship which might be recited in the claims. This position is untenable.

MPEP § 2112 states that “[t]he fact that a certain result or characteristic may occur or be present in the prior art is not sufficient to establish the inherency of that result or characteristic.” MPEP § 2112 further states that “[i]n relying upon the theory of inherency, the examiner must provide a basis in fact and/or technical reasoning to reasonably support the determination that the allegedly inherent characteristic necessarily flows from the teachings of the applied prior art.” The examiner has not provided a sufficient factual basis to prove inherency.

Applicants note that processors may have a multitude of possible architectures including different functional components, different structural components, and different interconnections between such components. Eggers itself describes many different possible functional arrangements for an SMT processor. However, it is not inherent in any of the SMT processor arrangements mentioned in Eggers that any particular component is coupled to any other particular component.

Even assuming, for the sake of argument, that Eggers described the recited first and second instruction fetch units and a multi-thread scheduler unit, it is not enough that such units might be coupled to each other. In order to rely on inherency, the Examiner must prove that in some SMT processor described by Eggers that such units must be coupled to each other. It is simply a truism that in some SMT processor architectures it might be possible that the scheduler unit is not coupled to first and second instruction fetch units. Accordingly, it does not necessarily follow such units are coupled to each other and the office action’s reliance on inherency fails, and the rejection should be reversed.

Claim 1, taken in its entirety and read as a whole, recites structural components and interconnections between those structural components, including, among other things, a multi-thread scheduler unit coupled to said first instruction fetch unit and said second instruction fetch unit. Eggers, which fails to identically describe even the first and second instruction fetch units or the recited multi-thread scheduler unit, further fails

to teach or suggest a multi-thread scheduler unit coupled to said first instruction fetch unit and said second instruction fetch unit, inherently or otherwise.

Claim 1 further recites an execution unit coupled to said scheduler unit, wherein said execution unit is to execute a first active thread and a second active thread. The final office action, for the first time, clarified that the rejection relies on page 13, left column, lines 15, 25, 34, and 51-52 for the recited execution unit.

Applicants note that the cited portions do not describe any execution unit. Rather, the cited portion only generally describes that instructions are executed. In any event, applicants again note that Eggers does not appear to describe any particular structure for the SMT processor alluded to therein. Rather, Eggers appears to be an academic paper describing proposals for processor designers to further investigate SMT techniques to improve multithreading processors. The cited portion fails to describe an execution unit coupled to a scheduler unit. The Examiner appears to again be relying on an unreasonably broad interpretation of the term "coupled." Moreover, it is not inherent, in any SMT processor described by Eggers, that the execution unit be coupled to the scheduler unit. In some SMT processor architectures, the execution unit might not be coupled to the scheduler unit. Accordingly, the Examiner's further reliance on inherency fails and the rejection should be reversed.

Claim 1 further recites a register file coupled to said execution unit, wherein said register file is to switch one of said first active thread and said second active thread with a first inactive thread. The office action relies on pages 14-15, particularly the section entitled "Register file and pipeline," for allegedly identically disclosing this claim recitation. Applicants are unable to identify any description in the cited portion that discloses a register file coupled to an execution unit or that the registers described in the "Register file" section are used to switch either a first or second active thread with an inactive thread. The office action goes on to identify actions described in Eggers as occurring at the fetch unit, not the register file. Namely, that the fetch unit may select

two of an available eight threads for decoding. Applicants do not understand the Examiner's analysis with respect to how the fetching of instructions by Eggers' fetch unit teaches or suggests anything whatsoever in connection with the operation of Eggers' register file. The office action appears to be improperly conflating these two distinct components of Eggers in order to read on the claims. In any event, selecting among available threads for execution in an SMT is different from and does not teach or suggest the recited switching of an active thread with an inactive thread.

With respect to numbered paragraph 11 in the final office action, the office action asserts that the Examiner is not conflating the register file with the fetch unit. However, the office action does not clarify what the Examiner is relying on and persists in referring to actions occurring at the fetch unit. The final office action asserts that Eggers describes "when the fetch unit switches one of said first active thread and said second active thread with a first inactive thread, the register files being accessed correspond to each particular thread being executed." This is a factually inaccurate representation of the teaching of Eggers made without citation to any corresponding portion of Eggers. Applicants are unable to identify any portion of Eggers which might correspond to such teaching. The final office action further asserts that "[s]o when the two threads are being executed, the corresponding register files for the two threads have been switched to and are being accessed. The register files effectively switch, along with the fetch unit, to one of said first active thread and said second active thread with a first inactive thread." Again this appears to be a factually inaccurate representation of the teaching of Eggers made without citation to any corresponding portion of Eggers. Applicants are unable to identify any portion of Eggers which might correspond to such teaching. In any event, a register file that switches with the execution of the threads is different from, and does not teach or suggest the recited register file which is to switch one of said first active thread and said second active thread with a first inactive thread.

By way if background, some embodiments of the present invention may relate to a multithreading processor which exhibits both simultaneous multithreading (SMT)

capability and switch on event multithreading (SoEMT) capability. In contrast, Eggers describes only multithreading processors having either SMT capability or SoEMT capability. Accordingly, Eggers cannot possibly anticipate the present claims or render the present claims unpatentable. In fact, Eggers teach away from such a combination of functionality because Eggers clearly suggests using SMT capabilities instead of SoEMT capabilities. According to Eggers, "SMT was able to get higher instruction throughput and greater program speedups than the fine-grained multithreading processor" (i.e. an SoEMT processor). See Eggers, page 17, section entitled "SMT vs. fine-grained multithreading."

With respect to paragraph 12 of the final office action, the foregoing, as explicitly stated, is provided by way of background.

For example, claim 1 recites an execution unit to execute a first active thread and a second active thread (e.g. SMT capability), and a register file to switch one of said first active thread and said second active thread with a first inactive thread (e.g. SoEMT capability). Among other things, Eggers fails to teach or suggest a multithreading processor with both these capabilities, as recited in claim 1.

Applicants note that the final office action does not address the foregoing argument, which does relate to what is recited in the claim. Accordingly, applicants consider that the Examiner's failure to answer this traversal is a concession of this argument and the rejection should be reversed.

The office action improperly relies on the description of the how Eggers' fetch unit operates for allegedly identically disclosing claim recitations relating to switching active threads. However, the operation of the fetch unit in Eggers does not have anything to do with active thread switching. As acknowledged by the Examiner in the office action, active threads are those threads which are executing. Eggers' fetch unit selects among various inactive instructions for execution. However, at the time of the instruction fetch, such instructions would not be considered active threads by those

skilled in the art. Rather, the selected instructions become active at the time of execution. The cited portion of Eggers, namely page 14, left hand column, describes that during the next instruction fetch cycle, a new instruction may be fetched to replace the now inactive instruction, but Eggers does not teach or suggest switching an active thread with a new thread.

For the foregoing reasons, in particular because Eggers fails to teach or suggest the various recitations noted above in claim 1, claim 1 is not anticipated by and is patentable over Eggers. Claims 2-8 depend either directly or indirectly from claim 1 and are likewise patentable.

#### Claim 2

With respect to claim 2, applicants note that Eggers does not describe how the "hardware contexts for eight threads" are otherwise interconnected. In any event, they do not appear to be coupled to the disclosed "register file." Applicants note that the office action relies on an unreasonably broad interpretation of the term "coupled" and, given the multiplicity of possible architectures, it is not inherent that the "hardware contexts" in Eggers are coupled to the register file in Eggers. Accordingly, claim 2 is separately patentable over Eggers.

#### Claim 4

With respect to claim 4, applicants note that the office action relies on Page 14, first paragraph, which does not appear to mention any decoders. The only mention of decoding applicants can identify is in the paragraph spanning the left and right columns on Page 14, which describes that Eggers single fetch unit selects subset of instructions from two threads for decoding. Accordingly, Eggers does not teach or suggest first and second decode units. As it is possible to use a single decode unit for two fetched instructions, it is not inherent that Eggers necessarily requires first and second decode

units. Accordingly, the Examiner's reliance on inherency fails and the rejection should be reversed.

The Examiner's comments with respect to "separate, independent, and distinct" are not understood. Claims must be interpreted to give meaning to every claim recitation. If the recited 'second instruction decode unit' was identical with the recited 'first instruction decode unit,' the claim recitation would be superfluous. Assuming for the sake of argument, that the single fetch unit in Eggers is coupled to a single decode unit, a single decode unit cannot possibly read on both the recited first and second decode units. Claim 4 further recites that the first decode unit is coupled between the first instruction fetch unit and the scheduler and that the second decode unit is coupled between the second instruction fetch unit and the scheduler. Because many different architectures are possible for any given SMT processor, it is not inherent that Eggers requires the recited structure. Accordingly, claim 4 is separately patentable over Eggers.

#### Claims 9 and 15

With respect to independent claims 9 and 15, the office action misconstrues the Eggers reference. Claims 9 and 15 each recite switching a first active thread with a third thread, if the third thread is ready to execute. The office action relies on the description of the how Eggers' fetch unit operates for allegedly identically disclosing this claim recitation. However, the operation of the fetch unit in Eggers does not have anything to do with active thread switching. As acknowledged by the Examiner in the office action, active threads are those threads which are executing. Eggers' fetch unit selects among various inactive instructions for execution. However, at the time of the instructions fetch, such instructions would not be considered active threads by those skilled in the art. Rather, the selected instructions become active at the time of execution.

Accordingly, the office action is in error by relying upon operations occurring at Eggers' fetch unit to read upon either of the detecting or switching recitations, which relate to an active thread. As noted above, the cited portion of Eggers describes only a

multithreading processor having SMT capability, not SoEMT capability. Accordingly, the only action Eggers teaches or suggests in connection with active threads during an execution cycle is the simultaneous execution of the active threads. When a stalling event occurs, the active thread becomes inactive and stops executing. The cited portion of Eggers, namely page 14, left hand column, describes that during the next instruction fetch cycle, a new instruction may be fetched to replace the now inactive instruction, but Eggers does not teach or suggest switching an active thread with a new thread after detecting a stalling event in the active thread.

With respect to paragraph 17 of the final office action, the foregoing is clearly discussing what is or is not described by Eggers, and not arguing features of the claims. In order to establish a *prima facie* case of obviousness, the Examiner must correctly identify what is taught by the reference. The foregoing is simply pointing out the errors in the Examiner's analysis. Applicants note that the office action does not take issue with the substance of applicants' argument. Namely, that the relied upon portion of the Eggers reference does not describe switching an active thread.

With respect to paragraph 18 of the final office action, the Examiner's response fails to address applicants' traversal. Applicants note that the office action does not take issue with the substance of applicants' argument. Namely, that the relied upon portion of the Eggers reference does not describe switching an active thread.

Because, among other things, Eggers fails to teach or suggest the recited switching said first active thread with a third thread, if the third thread is ready to execute, claims 9 and 15 are not anticipated by and are patentable over Eggers. Claims 10-14 depend from claim 9 and are likewise patentable. Claims 16-20 depend from claim 15 and are likewise patentable.

II. The rejection of claims 5-8 under 35 U.S.C. § 103(a) as being unpatentable over Eggers is in error and should be reversed.

Claim 5

Claim 5 depends from claim 1 and is therefore patentable for at least the reasons given above in connection with claim 1.

With respect to claim 5, applicants again note that Eggers does not describe any particular structure for the SMT processor proposed therein. Moreover, Eggers at most describes a single fetch unit which can fetch two or more instructions in a given cycle for decoding by a single decode unit. This is different from and does not teach or suggest the structure of multiple (e.g. first through fourth) instruction fetch units as recited in claim 5. Applicants again note that Eggers describe only partitioning the instruction bandwidth, not partitioning the fetch unit. Accordingly, claim 5 is separately patentable over Eggers.

Claim 6

Claim 6 depends from claim 5 and is therefore patentable for at least the reasons given above in connection with claims 1 and 5.

With respect to claim 6, applicants note that the register file mentioned in Eggers appears to have a different structure and function as compared to the recited four way register file. The final office action asserts that “a register file can have any number of ways.” However, not every register file structure is suitable for structuring with multiple ways. Merely changing the size of the register file described in Eggers still does not read on the recited four way register file. Applicants are not arguing any feature not recited in claim 6. Accordingly, claim 6 is separately patentable over Eggers.

Claim 7

Claim 7 depends from claim 6 and is therefore patentable for at least the reasons given above in connection with claims 1, 5, and 6.

With respect to claim 7, the office action again appears to improperly conflate the functions of Eggers' fetch unit with Eggers' register file. The Examiner does not unambiguously identify whether the rejection relies on Eggers' fetch unit or Eggers' register file for the recited switching. In the absence of such unambiguous analysis, the office action is in error as to what is taught or suggested by Eggers and fails to establish a *prima facie* case of obviousness. Accordingly, claim 7 is separately patentable over Eggers.

Claim 8

Claim 8 depends from claim 7 and is therefore patentable for at least the reasons given above in connection with claims 1, 5, 6, and 7.

With respect to claim 8, applicants again note that Eggers does not describe any particular structure for the SMT processor proposed therein. Moreover, Eggers at most describes a single fetch unit which can fetch two or more instructions in a given cycle for decoding by a single decode unit. This is different from and does not teach or suggest the structure of multiple (e.g. first through fourth) decode units as recited in claim 8. Accordingly, claim 8 is separately patentable over Eggers.

CONCLUSION

In view of the foregoing, favorable reconsideration and reversal of the rejections is respectfully requested. Early notification of the same is earnestly solicited. If there are any questions regarding the present application, the Examiner and / or the Board is invited to contact the undersigned attorney at the telephone number listed below.

Respectfully submitted,

July 26, 2005

Date

/Paul E. Steiner/  
Paul E. Steiner  
Reg. No. 41,326  
(703) 633 - 6830

Intel Americas  
LF3  
4030 Lafayette Center Drive  
Chantilly, VA 20151

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to:  
Commissioner for Patents, P.O. Box 1450 Alexandria, VA 22313-1450

On: July 26, 2005

Signature:

  
Katherine Jennings

**(viii) *Claims appendix.***

1. A multi-threading processor, comprising:
  - a first instruction fetch unit and a second instruction fetch unit;
  - a multi-thread scheduler unit coupled to said first instruction fetch unit and said second instruction fetch unit;
  - an execution unit coupled to said scheduler unit, wherein said execution unit is to execute a first active thread and a second active thread; and
  - a register file coupled to said execution unit, wherein said register file is to switch one of said first active thread and said second active thread with a first inactive thread.
2. A multi-threading processor as recited in claim 1, further comprising an on deck context unit coupled to the register file, wherein said on deck context unit is to maintain the first inactive thread and a second inactive thread.
3. A multi-threading processor as recited in claim 2, wherein said register file is to switch one of the first active thread and the second active thread with the second inactive thread.
4. A multi-threading processor as recited in claim 3, further comprising:
  - a first instruction decode unit coupled between the first instruction fetch unit and the scheduler unit; and
  - a second instruction decode unit coupled between the second instruction fetch unit and the scheduler unit.

5. A multi-threading processor as recited in claim 1, wherein the scheduler unit is a four thread scheduler unit, further comprising:
  - a third instruction fetch unit coupled to said four thread scheduler unit; and
  - a fourth instruction fetch unit coupled to said four thread scheduler unit.
6. A multi-threading processor as recited in claim 5, wherein said register file is a four way register file.
7. A multi-threading processor as recited in claim 6, wherein said register file is to switch one of the first active thread and the second active thread with a second inactive thread.
8. A multi-threading processor as recited in claim 7, further comprising:
  - a third instruction decode unit coupled between the third instruction fetch unit and the four thread scheduler unit; and
  - a fourth instruction decode unit coupled between the fourth instruction fetch unit and the four thread scheduler unit.
9. A method for switching threads in a multi-threading processor, comprising:
  - fetching a first active thread and a second active thread;
  - detecting a stalling event in said first active thread; and
  - switching said first active thread with a third thread, if the third thread is ready to execute.

10. A method for switching threads as recited in claim 9, further comprising executing the third thread and said second active thread.

11. A method for switching threads as recited in claim 9, further comprising detecting a stalling event in the second active thread.

12. A method for switching threads as recited in claim 11, further comprising switching said second active thread with a fourth thread, if the fourth thread is ready to execute.

13. A method for switching threads as recited in claim 12, further comprising executing the third thread and the fourth thread.

14. A method for switching threads as recited in claim 9, further comprising executing the second active thread, if the third thread is not ready to execute.

15. A set of instructions for switching threads residing in a storage medium, said set of instructions capable of being executed by a processor, comprising:

fetching a first active thread and a second active thread;  
detecting a stalling event in said first active thread; and  
switching said first active thread with a third thread, if the third thread is ready to execute.

16. A set of instructions for switching threads as recited in claim 15, further comprising executing the third thread and said second active thread.

17. A set of instructions for switching threads as recited in claim 15, further comprising detecting a stalling event in the second active thread.

18. A set of instructions for switching threads as recited in claim 17, further comprising switching said second active thread with a fourth thread, if the fourth thread is ready to execute.

19. A set of instructions for switching threads as recited in claim 18, further comprising executing the third thread and the fourth thread.

20. A set of instructions for switching threads as recited in claim 15, further comprising executing the second active thread, if the third thread is not ready to execute.

**(ix) Evidence appendix.**

None.

**(x) Related proceedings appendix.**

None.